A Multi-Resolution Nonlinear Mapping Technique For Design and Analysis Applications

Minh Q. Phan
Department of Mechanical and Aerospace Engineering
Princeton University
Princeton, NJ 08544

Final Technical Report for
NASA Grant NAG-1-1843

A Multi-Resolution Nonlinear Mapping Technique For Design and Analysis Applications

Minh Q. Phan
Department of Mechanical and Aerospace Engineering
Princeton University
Princeton, NJ 08544

Abstract

This report describes a nonlinear mapping technique where the unknown static or dynamic system is approximated by a sum of dimensionally increasing functions (one-dimensional curves, two-dimensional surfaces, etc.). These lower dimensional functions are synthesized from a set of multi-resolution basis functions, where the resolutions specify the level of details at which the nonlinear system is approximated. The basis functions also cause the parameter estimation step to become linear. This feature is taken advantage of to derive a systematic procedure to determine and eliminate basis functions that are less significant for the particular system under identification. The number of unknown parameters that must be estimated is thus reduced and compact models obtained. The lower dimensional functions (identified curves and surfaces) permit a kind of "visualization" into the complexity of the nonlinearity itself.

Introduction

It has been widely recognized nonlinear system identification is an important problem from both theoretical and practical perspectives. Unlike the case of linear system identification where systematic and extensive results have been achieved within the unifying framework of linear system theory, there is no such parallel development for

the case of nonlinear systems. In some cases, system identification simply refers to the process of using the known structure of some analytical model of a system to estimate its unknown physical parameters from measured input-output data. In other cases, it is not possible or practical to derive a physical model due to the lack of physical insights or model complexity. Generation of a black-box model from measured data may then be the only option. Parameters in such a black-box model do not have any physical interpretation. Whether or not a black-box model can capture the underlying nonlinearity of the physical system depends on (1) the class of nonlinear systems that a particular type of black box model is capable of representing, and (2) the "tunability" of the parameters within the black box structure. This assumes of course that sufficient amount of data is available for model identification.

It is easy to see why identifying a high dimensional nonlinear function of arbitrary complexity can be quite difficult by examining the amount of data that is needed to sample to input space versus the amount of data that one normally has available. For 10-input problem, if the grid size is 10 for each input variable, one would need 10^{10} or 10 billion data points to sample evenly the input space. Obviously, this is beyond any reasonable amount of data that one may actually have in practice. One hundred thousand points (a rather large number by today's standard) represents only a tiny fraction (0.001%) of that amount. Additional assumptions such as smoothness or low contribution of higher-order terms must be made to bring the problem down to an intuitively realistic level. Fortunately this is often the case for many physical systems. Otherwise high dimensional problems can be easily unmanageable.

As far as black-box models are concerned, perhaps the most studied one in the past is the Volterra series which has found applications in both identification and control [1]. A common deficiency of this model is the number of higher-order terms that must be retained for high-fidelity representation can be impracticably large. In recent years, the neural networks have emerged as the most viable approach for nonlinear system identification [2]. Theoretically, it has been proven that any nonlinear function can be represented by a multi-layer feedforward neural network of sufficient complexity [3].

However, knowing such a network exists is not the same as actually finding it. The standard multi-layer feedforward structure is a cascaded functional, e.g., functions of functions, each of which contains parameters to be tuned. Due to the network structure, these parameters relate to the input-output data nonlinearly. Training such a network is essentially a nonlinear optimization problem. Typically, it is solved using a gradient-based iterative scheme such as the well-known back propagation algorithm or its variants. Remarkably, numerous successes with such training schemes have been reported in the literature. However, it has also been recognized that the network training process may be very time-consuming, and susceptible to local minima. One may argue that the feedforward structure that gives the network its strength in the theoretical ability to model general nonlinear functions is also the source of its weakness in practice. It should be mentioned that a class of neural networks known as radial basis function networks aims at making the training problem linear [4]. This type of network models a nonlinear system as a linear combination of known basis functions (e.g., Gaussian functions) with unknown coefficients to be determined. In two-dimensional problems where it is easy to visualize, the nonlinear surface is defined by a number of hills and valleys each of which can be modeled by a suitably chosen basis function of appropriate size and at appropriate location. While the "optimal" location of the centers can be handled within the framework of linear theory, the "width" of the Gaussian basis functions (which influences into the compactness of the nonlinear model) must be specified in advance. Optimizing the widths as well as locations will result in a nonlinear optimization problem, and this will negate much of the primary advantage of the linear structure in the first place.

In this work, we explore a nonlinear mapping approach where the nonlinear function is approximated by sum of dimensionally increasing functions. Recently, Ref. [5] developed a procedure to construct such an expansion where each of the lower-dimensional functions is expressed in terms of various integrals of the original function. This construction is significant in that it shows that the lower-dimensional functions are more significant than the higher-dimensional ones which correspond to the less important higher-order statistics (e.g., mean and variance are important but variance of variance is

rarely used). In a system identification problem, the original function is not known and needs to be estimated, thus it is not possible to use this expansion directly. Although Monte Carlo integration may be used to obtain these integrals numerically from input-output data, the amount of needed data is too large to be practical. A variant of this expansion involves function evaluation about a reference point as opposed to function integration over the input domain [6]. To construct such an expansion, however, one must be able to generate specialized input data by holding most input variables fixed at the reference values while varying others. In practice, this is not always possible especially when modeling dynamic systems where some of the "input" variables are actually time-delayed output values. It also precludes the use of excitation data that sample the input domain randomly. More importantly, minimizing the prediction error is not a stated goal in these developments although such a goal is obviously natural from system identification point of view. Despite these limitations, it has been recently observed that expansions of this type, involving only first and sometimes second-order terms, can capture the underlying nonlinear relationship rather well. Two such applications are found in chemical kinetics [6] and materials design [7]. The notion of "order" in this context is related to the dimension of the constitutive components of an expansion the dimension space. It is not related to the "power" in a power series expansion as in the case of a Taylor series. Even "first-order" terms can have arbitrary nonlinearity.

Motivated by this observation, we carry this line of thinking one step further by actually determining the lower-dimensional expansions that minimize the prediction error of the model itself. Furthermore, it is natural to model each of these first and second-order expansions by basis functions. In so doing, the parameter identification step becomes linear while the system remains nonlinear, thus eliminating difficulties inherent in any nonlinear iterative approaches. We are essentially trading the compactness of the feedforward neural network structure for linearity in the parameter identification step. In return, the number of parameters that must be determined can be large. To handle this problem, we will show how to take advantage of the linear feature to reduce the number of parameters that must be calculated. This is achieved by eliminating nonsignificant

basis functions. Furthermore, the basis functions are arranged in a multi-resolution fashion consisting of dilated (or compressed) and translated versions of a single mother basis function. This is similar to the way wavelets are constructed [8]. The multi-resolution feature allows one to generate approximate models at different level of details and this can be matched up with the available data. Furthermore, it is also numerically efficient in that a higher resolution model can be generated without undoing most of the calculations in obtaining the lower resolution model. It should also be mentioned here that by decomposing the unknown system in terms of curves and surfaces, we have a way to "visualize" the nonlinearity itself. This is a feature not found in conventional nonlinear mapping techniques.

Dimensionally Increasing Function Expansion

The basic concept is to decompose a real-valued function of several input variables into a sum of dimensionally increasing functions. Such an expansion for $y = f(x_1, x_2, ..., x_n)$ has the form

$$f(x_1, x_2, ..., x_n) = f_0 + \sum_{i=1}^{n} f_i(x_i) + \sum_{1 \le i \le j}^{n-1} \sum_{j \le n}^{n} f_{i,j}(x_i, x_j) + \cdots + f_{1,2,...,n}(x_1, x_2, ..., x_n) \qquad (1)$$

where f_0 is a constant, $f_i(x_i)$ is a function of the input variable x_i alone, $f_{ij}(x_i, x_j)$ is a function of the input variables x_i, x_j alone, etc... The last term involves all input variables thus making (1) an identity and not an approximation. It has been shown in Ref. [5] that if one imposes the condition that the integral of each of the summands in (1) with respect to any of their own variables is zero, i.e.,

$$\int_0^1 \int_0^1 \cdots \int_0^1 f_{1,2,...,s}(x_1, x_2, ..., x_s) dx_k = 0, \quad 1 \le k \le s \qquad (2)$$

where every input variable is normalized to fall in the range between 0 and 1, $0 \le x_i \le 1$,

then it follows that the summands on the right hand side of (1) are orthogonal to each other. Orthogonality here means that the integral of the product of any two (different) summands with respect to all their own variables is zero over the relevant domain, i.e.,

$$\int_0^1\int_0^1\cdots\int_0^1 f_{i_1,\ldots,i_s}(x_{i_1},\ldots,x_{i_s})f_{j_1,\ldots,j_t}(x_{j_1},\ldots,x_{j_t})dx_{i_1}\ldots dx_{j_t}=0,\quad (i_1,\ldots,i_s)\neq(j_1,\ldots,j_t) \tag{3}$$

Under these imposed conditions, there exists a unique expansion of (1) for any integrable function in the domain of interest. Specifically,

$$\begin{aligned} f_0 &= \int_0^1\int_0^1\cdots\int_0^1 f(x)dx \\ f_i(x_i) &= \int_0^1\int_0^1\cdots\int_0^1 f(x)\frac{dx}{dx_i} - f_0 \\ f_{i,j}(x_i,x_j) &= \int_0^1\int_0^1\cdots\int_0^1 f(x)\frac{dx}{dx_i dx_j} - f_i(x_i) - f_j(x_j) - f_0, \text{ etc...} \end{aligned} \tag{4}$$

For simplicity, x is used to denote all input variables. If the original function known then the lower dimensional functions can be directly derived using (4). If the original function is not known but data is available then they can be evaluated by Monte Carlo integration directly from input-output data. From a practical point of view, however, this is not preferable because of the large amount of data needed. A variant of this expansion replaces function integration by function evaluation at various cuts about some reference point,

$$f(x_1,x_2,\ldots,x_n)=\tilde{f}_0+\sum_{i=1}^{n}\tilde{f}_i(x_i)+\sum_{1\le i\le j\,j\le n}^{n-1}\sum^{n}\tilde{f}_{i,j}(x_i,x_j)+\cdots+\tilde{f}_{1,2,\ldots,n}(x_1,x_2,\ldots,x_n) \tag{5}$$

where the summands are obtained by varying one or more variables while keeping others fixed,

$$\bar{f}_0 = f(\bar{x}_1, \bar{x}_2, ..., \bar{x}_n)$$
$$\bar{f}_i(x_i) = f(\bar{x}_1, ..., x_i, ..., \bar{x}_n) - \bar{f}_0 \qquad (6)$$
$$f_{i,j}(x_i, x_j) = f(\bar{x}_1, ..., x_i, ..., x_j, ..., \bar{x}_n) - \bar{f}_i(x_i) - \bar{f}_j(x_j) - f_0, \quad \text{etc.}$$

The above construction does not necessarily minimize the approximation error, and it also requires special data where certain input variables are varied while others fixed at the reference values. Despite these limitations, it was still found to be remarkably effective in modeling many physical phenomena [6,7]. Our present goal is not in the derivation of analytical expressions that minimize the approximation error given a known nonlinear function. Instead the nonlinear function is unknown and our goal is in finding efficient ways to extract the dimensionally increasing functions from input-output data itself (an identification problem), and to do so in such a way that the approximating error is minimized for the available identification data record.

Basis Function Representation

Let us now consider a modified version of Eq. (1),

$$f(x_1, x_2, ..., x_n) = g_0 + \sum_{i=1}^{n} \alpha_i x_i + \sum_{i=1}^{n} g_i(x_i) + \sum_{1 \le i \le j}^{n-1} \sum_{j \le n}^{n} g_{i,j}(x_i, x_j) + e(x_1, x_2, ..., x_n) \qquad (7)$$

where $e(x_1, x_2, ..., x_n)$ denotes the error of the representation to be minimized in the identification step. We have separated the linear terms from the rest of the expansion. The reasons for this separation are two-fold. First, it provides for an explicit separation of the linear and nonlinear terms. Second, the nonlinear terms will be approximated later by a finite number of basis functions, each of which can have a different but finite resolution. The linear terms, on the other hand, have "infinite" resolution in the sense that any linear function is uniquely defined by a constant (bias) and a coefficient (slope). Each of the nonlinear terms in Eq. (7) will be modeled by basis functions as follows,

$$g_i(x_i)=\sum_{k=1}^{N_1}\beta_k^{(i)}\phi_k(x_i),\qquad g_{ij}(x_i,x_j)=\sum_{k=1}^{N_2}\gamma_k^{(ij)}\varphi_k(x_i,x_j) \tag{8}$$

Given a set of input-output data of sufficient length, the identification problem then is finding $g_0,\alpha_i,\beta_k^{(i)},\gamma_k^{(ij)}$ such that the fitting error $e(x_1,x_2,...,x_n)$ is minimized over the entire data record. The input-output equation can be arranged as,

$$y=p^T\psi+e \tag{9}$$

where p is a column vector of the unknown parameters containing a constant g_0, linear coefficients $\{\alpha_i\}$, first-order nonlinear coefficients $\{\beta_k^{(i)}\}$, and second-order nonlinear coefficients $\{\gamma_k^{(ij)}\}$,

$$p^T=\left[g_0,\{\alpha_i\},\{\beta_k^{(i)}\},\{\gamma_k^{(ij)}\}\right] \tag{10}$$

where $\{\alpha_i\},\{\beta_k^{(i)}\},\{\gamma_k^{(ij)}\}$ consists of

$$\begin{gathered}\{\alpha_i\}=[\alpha_1,\alpha_2,...,\alpha_n],\qquad \{\beta_k^{(i)}\}=\left[\beta_i^{(1)},...,\beta_{N_1}^{(1)},\cdots,\beta_i^{(n)}...,\beta_{N_1}^{(n)}\right]\\ \{\gamma_k^{(ij)}\}=\left[\gamma_1^{(12)},...,\gamma_{N_2}^{(12)},\cdots,\gamma_1^{(n-1,n)},...,\gamma_{N_2}^{(n-1,n)}\right]\end{gathered} \tag{11}$$

The corresponding "input" vector is

$$\psi^T=\left[1,x_1,...,x_n,\{\phi_k(x_i)\},\{\varphi_k(x_i,x_j)\}\right] \tag{12}$$

where

$$\{\phi_k(x_i)\}=\left[\phi_1(x_1),...,\phi_{N_1}(x_1),\phi_1(x_2),\cdots,\phi_{N_1}(x_2),...,\phi_1(x_n),...,\phi_{N_1}(x_n)\right]$$

(13)

$$\left\{\varphi_k(x_i, x_j)\right\} = \left[\varphi_1(x_1, x_2), ..., \varphi_{N_2}(x_1, x_2), \cdots, \varphi_1(x_{n-1}, x_n), ..., \varphi_{N_2}(x_{n-1}, x_n)\right]$$

For clarity, it is noted here that the term "first-order" or "second-order" as used in the present context refers to the dimension of the associated functions. Even "first-order" terms can have arbitrary nonlinearity. They are not to be confused with the same terminology used in a Taylor series expansion where the notion of "order" is attached to the power of the expansion (e.g., first-order terms are linear, second-order terms are quadratic, and so on).

Identification of Basis Function Coefficients

Once the unknown parameters are arranged in the form given in (9), the identification problem is straightforward. The unknown coefficients in p can be estimated in recursive mode or in batch mode. Recursive computation is appropriate when the identification is to be carried out in real time for on-line identification. When the number of unknowns is large, recursive computation is also appropriate because by operating on one data sample at time, the computation is memory efficient. For recursive computation, we have at our disposal a rather large number of available algorithms [9]. One obvious choice is the well-known recursive least-squares algorithm,

$$\hat{p}(k+1) = \hat{p}(k) + \frac{R(k)\psi(k)}{1 + \psi(k)^T R(k)\psi(k)}\left[y(k) - \hat{p}(k)^T \psi(k)\right]$$

$$R(k+1) = R(k) - \frac{R(k)\psi(k)\psi(k)^T R(k)}{1 + \psi(k)^T R(k)\psi(k)} \tag{14}$$

starting with an initial guess $\hat{p}(0)$ and any positive definite matrix $R(0)$. The index k denotes the k-th data sample in $y(k) = f(x_1(k), x_2(k), ..., x_n(k))$, and $\hat{p}(k)$ is the estimated parameter at the k-th iteration. This algorithm has initial rapid converge at the expense of additional computation to update the covariance matrix $R(k)$. A slower but much

simpler algorithm is the projection algorithm (also known as normalized least-mean squares algorithm),

$$\hat{p}(k+1) = \hat{p}(k) + \frac{\lambda_1 \psi(k)}{\lambda_2 + \psi(k)^T \psi(k)} \left[y(k) - \hat{p}(k)^T \psi(k) \right] \tag{15}$$

with $0 < \lambda_1 < 2$, $\lambda_2 > 0$. This algorithm involves only scalar product multiplications and there is no covariance matrix to compute. In terms of memory efficiency, it can handle large problems. For batch-type calculations, one simply form the following data matrices for a data record of ℓ samples long,

$$\begin{aligned} [y] &= [y(0), y(1), y(2), \ldots y(\ell-1)] \\ [\psi] &= [\psi(0), \psi(1), \psi(2), \ldots \psi(\ell-1)] \end{aligned} \tag{16}$$

Then the least-squares solution for p that minimizes the approximation error $[e]$ over the entire data record is simply

$$\hat{p} = [y][\psi]^+ = [y][\psi]^T \left([\psi][\psi]^T \right)^+ \tag{17}$$

where $(.)^+$ denotes the pseudo-inverse operation. This computation is best handled via the singular value decomposition of $[\psi]$ or $[\psi]^T[\psi]$.

Construction of Multi-Resolution Basis Functions

Next, we address the issue of how the basis functions should be structured for use in the above calculation. The multi-resolution strategy involves basis functions that are compressed (or dilated) and translated versions of a single mother basis function here taken to be a Gaussian function. This strategy is adopted with the following objectives in mind. First, it allows the identification of the coarse feature of the underlying nonlinear

relationship first, followed by additional refinement when higher resolution basis functions are used. Second, when upgrading to a higher resolution model from a lower resolution one, it avoids the reforming of input vectors associated with the lower resolution model, thus making the identification numerically efficient. Third, this arrangement also facilitates more compact models through a model reduction process where the less relevant basis functions are eliminated and the parameters associated with them. In the following we describe one such arrangement of the basis functions that meet the above stated objectives, but other constructions are also possible.

As mentioned, the input variables are normalized so that they fall in the interval between 0 and 1. Let us now consider the first-order basis functions. At the first resolution level, we choose 3 basis functions each has a "radius" of 1/2 and centered at 0, 1/2, and 1, respectively

$$\phi_1(x_i) = e^{-(x_i)^2/(1/2)^2}, \quad \phi_2(x_i) = e^{-(x_i-1/2)^2/(1/2)^2}, \quad \phi_3(x_i) = e^{-(x_i-1)^2/(1/2)^2} \tag{18}$$

The resolution for the first level is 1/2. At the second resolution level, 2 additional basis functions are introduced, each of radius 1/4 and centered at 1/4 and 3/4, i.e.,

$$\phi_4(x_i) = e^{-(x_i-1/4)^2/(1/4)^2}, \quad \phi_5(x_i) = e^{-(x_i-3/4)^2/(1/4)^2} \tag{19}$$

With the first two levels, the resolution now is 1/4. The next resolution calls for 4 additional basis functions, each of radius 1/8 and centered at 1/8, 3/8, 5/8, 7/8, corresponding to a resolution of 1/8. This process continues, each time one level is added, one doubles the fineness of the division in the interval between 0 and 1.

In the same fashion, we can construct the second-order basis functions (surfaces). The starting resolution is 1/2, 9 basis surfaces are chosen, each of radius 1/2, and centered at the intersections of a 2-by-2 grid, i.e.,

$$\varphi_1(x_i,x_j)=e^{-(x_i)^2-(x_j-1)^2/(1/2^2)},\ \varphi_2(x_i,x_j)=e^{-(x_i-1/2^2)-(x_j-1)^2/(1/2^2)},\ \ldots \qquad (20)$$

The second level involves 16 additional basis surfaces, each of radius 1/4 and centered at the intersections of a 4-by-4 grid, and so on. In the construction of both first and second-order basis functions, both with a starting resolution of 1/2, the basis functions are centered at the intersections of the grid as opposed to the centers of each grid element. This is intentionally done so that the nonlinear function can be well approximated at the boundaries of the input domain.

Elimination of Less Significant Basis Functions

In a typical problem, the nonlinear system has a finite number of primary features, each of which involves only a certain number of basis functions. Although one casts a generically large "net" of basis functions, it is expected that only a fraction of them is relevant for a particular system. It would be advantageous, therefore, to determine and keep only the relevant basis functions. By identifying only the parameters associated with them, one now has a smaller identification problem to solve and the resultant model is more compact. Another benefit of reducing number of unknowns is that it makes the identification problem becomes better conditioned.

The determination and elimination of less significant basis functions can be handled in a systematic manner by orthogonalization of the rows of the input matrix $[\psi]$, each of which corresponds to a specific basis function. Note that each row of $[\psi]$ is made up of the values of a particular basis function evaluated at the available input data samples. Because the input data may not span the input space evenly, these rows are not necessarily orthogonal to each other even if the basis functions are orthogonal. The rows of $[\psi]$ can be easily orthogonalized (or orthonormalized for convenience) by a linear transformation A,

$$[\overline{\psi}] = A[\psi] \tag{21}$$

such that $[\overline{\psi}][\overline{\psi}]^T = I$. The procedure can be performed one row at a time by a scheme such as the well-known Gram-Schmidt orthogonalization. Because orthogonalization does not change the space spanned by the rows of $[\psi]$, the least-squares error associated with the new coefficients in the new space is identical to the least-squares error associated with the old coefficients, i.e.,

$$\begin{aligned}
[\overline{\hat{y}}] &= [y][\overline{\psi}]^T \left([\overline{\psi}][\overline{\psi}]^T\right)^{-1}[\overline{\psi}] \\
&= [y][\psi]^T A^T \left(A[\psi][\psi]^T A^T\right)^{-1} A[\psi] \\
&= [y][\psi]^T \left([\psi][\psi]^T\right)^{-1}[\psi] = [\hat{y}]
\end{aligned} \tag{22}$$

In the above equation $[\overline{\hat{y}}]$ and $[\hat{y}]$ denote the least-squares fit of $[y]$ using the rows of $[\overline{\psi}]$ and of $[\psi]$, respectively. Furthermore, since the rows of $[\overline{\psi}]$ are orthogonal, the contribution of each row to explaining the data $[y]$ can be easily determined independently of other rows. Specifically, let $\overline{\psi}_i$ denote the i-th row of $[\overline{\psi}]$ associated with the i-th basis function then its contribution to explaining the data $[y]$ is

$$r_i = [y]\overline{\psi}_i^T \frac{\overline{\psi}_i \overline{\psi}_i^T}{[y][y]^T} \tag{23}$$

Note that in the above expression, $[y]\overline{\psi}_i^T$, $\overline{\psi}_i \overline{\psi}_i^T$, $[y][y]^T$ all are scalar products and can be easily computed. This ratio can then be used to rank order the basis functions in terms of their individual contribution to explaining the data. Given a desired tolerance level specified by the user, the basis functions that must be kept can be easily identified. Let $[\psi]_r$ denote a new input data matrix where only the significant basis functions are retained, and p_r the corresponding coefficients of the reduced-dimension model, the reduced least-squares problem to be solved is

$$[y] = p_r^T [\psi]_r + [e_r] \tag{24}$$

Note that the above procedure reduces the number of unknown parameters to be solved for without reducing the number of data points. Also the parameter reduction step is performed without having to explicitly solve for the coefficients of the full model first.

We now discuss the issue of the richness of the input data to prevent ill-conditioning in the identification. This problem can be addressed by examining that the rank of the "input" data matrix $[\psi]$ for the full model, or $[\psi]_r$ for the reduced model. Rank deficiency signals either the input signal is not rich enough, or the problem is over-parameterized and there is no unique solution. Testing the identification result on an independent set of data not used for identification will immediately reveal which is the case. If ill-conditioning is caused by the input data being not sufficiently rich then the identified model is not valid even though it may reproduce the identification data. If ill-conditioning is caused by over-parameterization then it just means that there exists more than one model with similar levels of accuracy. In this case the parameter reduction scheme presented here will eliminate the less relevant basis functions to produce a better-conditioned and compact model. If one insists on keeping the full model then numerical ill-conditioning can be eliminated by discarding the smaller singular values when computing the pseudo-inverse of $[\psi]$ or $[\psi]^T[\psi]$.

Relationship to Volterra Series and Neural Networks

In this section we discuss how the proposed mapping procedure is similar to and different from other well-known basis function approximation methods, notably the Volterra series which was well studied in the past, and the more recent Radial Basis Function (RBF) neural networks. We will also discuss how this mapping technique compares to the feedforward neural networks.

A Volterra series approximates a multi-variable nonlinear function in the form,

$$f(x_1, x_2, \ldots, x_n) = f_0 + f_1(.) + f_2(.) + f_3(.) + \ldots \tag{25}$$

where

$$f_1(.) = \sum_{i=1}^{\infty} h_i x_i \;,\; f_2(.) = \sum_{i=1}^{\infty}\sum_{j=1}^{\infty} h_{ij} x_i x_j \;,\; f_3(.) = \sum_{i=1}^{\infty}\sum_{j=1}^{\infty}\sum_{k}^{\infty} h_{ijk} x_i x_j x_k \tag{26}$$

Our expansion is similar to the Volterra expansion in that it also involves a summation of dimensionally increasing functions. In (5), however, these functions of lower dimensions can be arbitrary (in principle) whereas as in the Volterra series they are constrained to be products of the input variables. This distinction is an important one because in many cases the inefficiency in the Volterra series can be attributed to this *a priori* specification of the form of the lower dimensional functions.

A RBF neural network, on the other hand, models a nonlinear function in the form,

$$f(x_1, x_2, \ldots, x_n) = \lambda_0 + \sum_{i=1}^{N} \lambda_i \phi(\|\underline{x} - \underline{c}_i\|) \tag{27}$$

where $\underline{x} = [x_1, x_2, \ldots, x_n]^T$ denotes the input vector, $\phi(.)$ is an assumed (known) scalar function of the distance (radius) from the input vector $\underline{x}$ to a fixed center $\underline{c}_i$ taken from the input data set itself. Originally the centers are chosen arbitrarily from the input data set, but later systematic approaches for center selection are developed. One such procedure uses orthogonalized regression vectors for center selection [4]. However, since the data themselves are candidate centers in an RBF network, the center selection procedure starts with as many candidate centers as the number of data points, which can be very large. In our current approach, we can think of each local basis function as having its own "center". But these centers are not taken from the input data set as in the

case of an RBF network. The number of these basis functions depends only on the specified resolutions and not on the number of data points. Another distinction is that in the RBF expansion, each $\phi(.)$ is a function of all input variables as opposed to an expansion of functions of increasing dimensions. In view of this discussion, it is possible to marry the features of the RBF neural network architecture with our dimensionally increasing basis function expansion. This development will be addressed in future work.

We now discuss how the proposed technique compares to the well-known feedforward neural network. As mentioned, although it is known that a multi-layer feedforward network exists to represent a generic nonlinear system to any degree of accuracy, finding such a network is not a trivial matter. The main reason for this difficulty is that the network parameters are nonlinearly dependent on the input-output data. Typically, a gradient-based method such as the backpropagation algorithm is used to update the network parameters. Many factors can potentially cause the training to fail. The difficulty may be in the data, the selected network size, or in the training algorithm itself. Since a gradient based iterative method can only produce a local solution, one arrives at different solutions from different starting points of the iteration. Increasing the network size will surely aggravate this problem. Network training is often a trial-and-error process. When the training is unsuccessful one is left with a very unsatisfactory feeling as to what exactly went wrong. In contrast, a linear identification problem leaves little or no ambiguity regardless of size. In addition, by eliminating less significant basis functions, the actual number of parameters that must be solved is typically much smaller than that of the full model (see numerical example). The proposed strategy is justified when the definiteness of the linear approach outweighs the uncertainties of a nonlinear iterative approach. In such cases, one would rather solve a high dimensional linear problem in a single calculation than a low dimensional nonlinear problem iteratively.

Illustration

Consider a dynamic system characterized by two masses m_1, m_2 and three spring coefficients k_1, k_2, k_3,

$$m_1\ddot{x}_1(t)+(k_1+k_2)x_1(t)-k_2x_2(t)=0$$
$$m_2\ddot{x}_2(t)-k_2x_1(t)+(k_2+k_3)x_2(t)=0 \tag{29}$$

Suppose we are interested in extracting a relationship between the physical parameters defining the systems and its two natural frequencies, which are the two positive roots of

$$(m_1m_2)\omega^4-\{(k_1+k_2)m_2+(k_2+k_3)m_1\}\omega^2+\{(k_1+k_2)(k_2+k_3)-k_2^2\}=0 \tag{30}$$

Each mass coefficient is allowed to vary within the interval $1\le m_i\le 1000$ (Kg), and each stiffness coefficient $10\le k_i\le 10{,}000$ (N/m). A data set consisting of 500 uniformly random combinations of the 5 input values (normalized to fall in the interval between 0 and 1), and the corresponding frequencies is used for identification. Another independent data set of equal length is used to test the prediction quality of the identified model. For identification, resolutions of 1/8 for the first-order terms and 1/4 for the second-order terms are chosen. The full model at these resolutions has 301 parameters. Let us first examine the identification of the full model. An examination of 301 singular values of the input data matrix reveals that the majority of the singular values are "small" (202 singular values less than 1×10^{-9}, the largest being 11×10^{3}). This numerical ill-conditioning suggests that the model may be over-parameterized. Three options are available at this point: (1) proceed with the identification of the full model, (2) lower the resolutions, or (3) keep the specified resolutions but use the parameter reduction procedure to eliminate non-significant basis functions. To proceed with option (1), we eliminate the source of the numerical ill-conditioning by discarding the smaller singular values (say, 202 singular values less than 1×10^{-9}). The results are shown in Table 1 for 20 arbitrary out of the 500 random combinations of the input parameters of the testing set that is not used in the identification of the model. Also shown are results obtained with a first-order model alone. Overall, the prediction error is about 0.25% for the second-order model and 2.5% for the first-order model. The identification of these models takes about 1 minute on a desktop PC. Option (2) improves numerical conditioning by lowering

resolutions, but this may cause reduced accuracy. In view of the fact that the number of parameters in the full model is still relatively small (301), option (3) is preferred over option (2) because it allows for model reduction without lowering the specified resolutions. Figure 1 shows the quality of the prediction error versus the size of the identification model using the parameter reduction scheme described in this report. The prediction error is about 18% for a 2-parameter model, 11% for a 4-parameter model, 4.4% for a 17-parameter model, 1.6% for a 47-parameter model, 0.72% for a 65-parameter model, etc. Recall that the prediction error is 0.25% for the full 301-parameter model. To identify the coefficients of a reduced model, say for a 17-parameter model, we only need to invert (or perform a singular value decomposition of) a 17-by-17 matrix as opposed to working with a 301-by-301 matrix for the full model. Thus the size of the matrix inversion is significantly reduced through this parameter reduction scheme. This model consists of 1 constant bias and 5 linear terms (Figure 2), 5 first-order nonlinear curves (Figure 3), and 10 second-order nonlinear surfaces (Figures 4 and 5).

Conclusions

A data-based nonlinear mapping technique has been described in this report. This technique draws upon recent results in multi-dimensional function expansion from mathematical statistics in combination with the classical usage of basis functions for function approximation. This combination has a number of attractive features. First, it represents the nonlinear system as a sum of dimensionally increasing functions (curves, surfaces, etc.) each of which can be visualized so that insights into the underlying nonlinearities may be developed. Second, the basis expansion makes the parameter estimation step linear, thus permitting direct application of existing linear estimation tools to the nonlinear identification problem. The strategy avoids common pitfalls associated with a nonlinear iterative parameter estimation technique as in the case of training a multilayer feedforward neural network. Third, the basis functions are constructed in a multi-resolution fashion that allows the system to be identified at various levels of details. The upgrade from a lower resolution model to a higher one can be made numerically efficient with this setup. Fourth, embedded in the identification technique is a procedure

that determines and keeps only the significant basis functions. This process substantially reduces the actual number of parameters that must be identified, significantly enhances numerical robustness, and results in compact model representation. When implemented recursively, one has a way to model slowly time-varying nonlinear systems.

As in the case of a Volterra series or a typical neural network, extension to dynamic mapping is relatively straightforward through the use of time-delayed input and output values. Generally speaking, to represent a generic nonlinear dynamic model, it has been widely recognized that the nonlinear auto-regressive moving-average (NARMA) model is a good choice. This model states that the current output value is a nonlinear function of a finite number of past input and past output values. Treating each of the time-delayed output and input values as x_i, all previous results immediately apply. Another extension involves a scheme where key elements of the proposed technique are combined with the radial construction of the radial basis function neural networks. Application of this mapping technique to the control area also represents a natural next step. In short, the proposed nonlinear mapping method is useful whenever the definiteness of the linear calculation and the insights it provides outweigh the uncertainties associated with a nonlinear iterative approach. With rapidly advancing computing technology, it is expected that nonlinear identification along this direction will become more and more attractive.

References

[1] Doyle, F.J., Ogunnaike, B.A., and Person, R.K., "Nonlinear Model-Based Control Using Second-Order Volterra Models," *Automatica*, Vol. 31, No. 5, 1995, pp. 697-714.

[2] Narendra, K.S., and Parthasarthy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 1, No. 1, 1990, pp. 4-27.

[3] Zurada, J.M., *Introduction to Artificial Neural Systems*, Info Access Distribution Ltd., Singapore, 1992.

[4] Chen, S., Cowan, C.F.N., and Grant, P.M., "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks," *IEEE Transactions on Neural Networks*, Vol. 2, No. 2, 1991, pp. 302-309.

[5] Sobol, I., "Sensitivity Estimates for Nonlinear Mathematical Models," *Mathematical Modeling and Computational Experiments*, Vol. 1, 1993, pp. 407-414.

[6] Shorter, J., Ip, P.C., and Rabitz, H., "An Ultra-fast Chemistry Solver Using High Dimensional Model Representations," *in preparation.*

[7] Shim, K. and Rabitz, H., "Independent and Correlated Composition Behavior of the Energy Band Gaps for the $Ga_{\alpha}In_{1-\alpha}P_{\beta}As_{1-\beta}$ and $Ga_{\alpha}In_{1-\alpha}P_{\beta}Sb_{\gamma}As_{1-\beta-\gamma}$ Alloys," *Phys. Rev. B*, 58, 1998, pp. 1940-1946.

[8] Strang, G., and Nguyen, T., *Wavelets and Filter Banks*, Wellesley-Cambridge Press, 1997, pp. 174-186.

[9] Goodwin, G.C. and Sin, K.S., *Adaptive Filtering Prediction and Control*, Prentice-Hall, New Jersey, 1984, pp. 50-67.

Table 1: True and predicted natural frequencies for 20 random combinations of input parameters not used in the identification step.

True ω_1	Predicted ω_1 Second-Order	Predicted ω_1 First-Order	True ω_2	Predicted ω_2 Second-Order	Predicted ω_2 First-Order
5.1495	5.1505	5.1286	**9.3882**	9.3905	9.3489
2.8783	2.8784	2.9084	**5.2538**	5.2550	5.2123
3.9899	3.9901	3.9702	**9.0155**	9.0138	9.1353
2.8048	2.8055	3.0582	**9.0896**	9.0888	8.7835
2.1946	2.1945	2.1829	**4.1396**	4.1394	4.0593
3.1161	3.1172	3.0595	**9.1126**	9.1126	9.1179
3.1530	3.1525	3.1559	**5.8683**	5.8674	5.8742
2.5076	2.5075	2.4470	**4.3700**	4.3700	4.5038
5.1316	5.1307	5.1187	**9.3411**	9.3391	9.3658
1.3685	1.3691	1.3976	**5.2052**	5.2038	4.9504
2.9791	2.9789	2.9802	**5.4013**	5.3999	5.5130
3.0024	3.0020	2.9658	**5.2011**	5.1997	5.2513
2.2722	2.2682	2.3759	**5.4858**	5.4767	5.9023
2.1947	2.1943	2.2406	**3.8903**	3.8897	3.7795
3.7324	3.7326	3.7303	**10.2421**	10.2354	10.1782
5.0724	5.0844	5.0745	**9.1892**	9.2158	9.0430
3.0573	3.0576	3.0067	**5.7638**	5.7648	5.9476
4.0497	4.0498	4.0062	**8.8273**	8.8308	8.8807
3.1545	3.1545	3.0680	**5.4637**	5.4632	5.6371
2.7503	2.7504	2.7403	**4.7712**	4.7714	4.8378

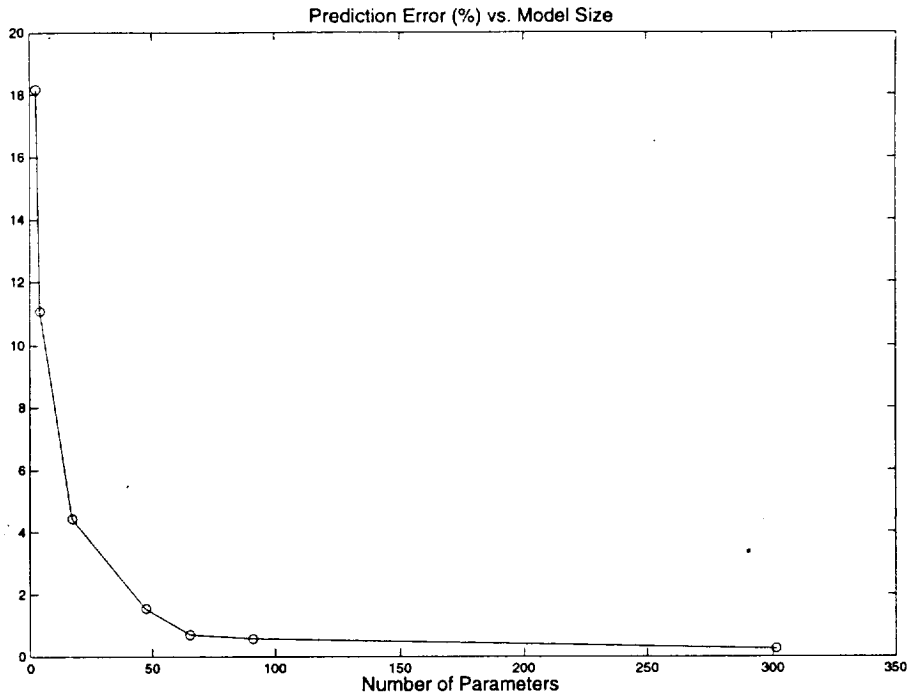

Figure 1: Prediction error vs. model size.

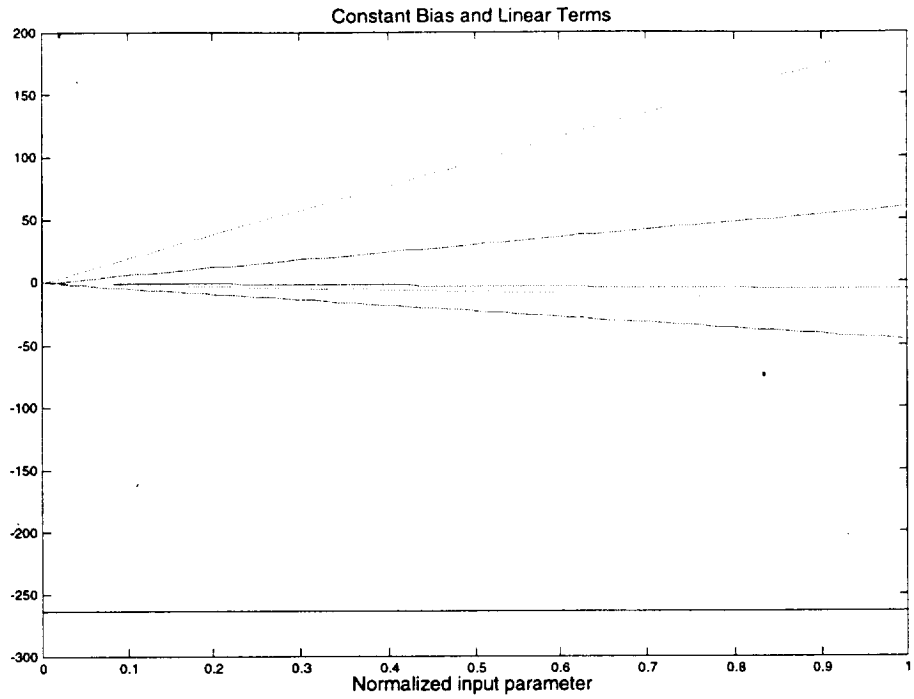

Figure 2: Constant and linear terms.

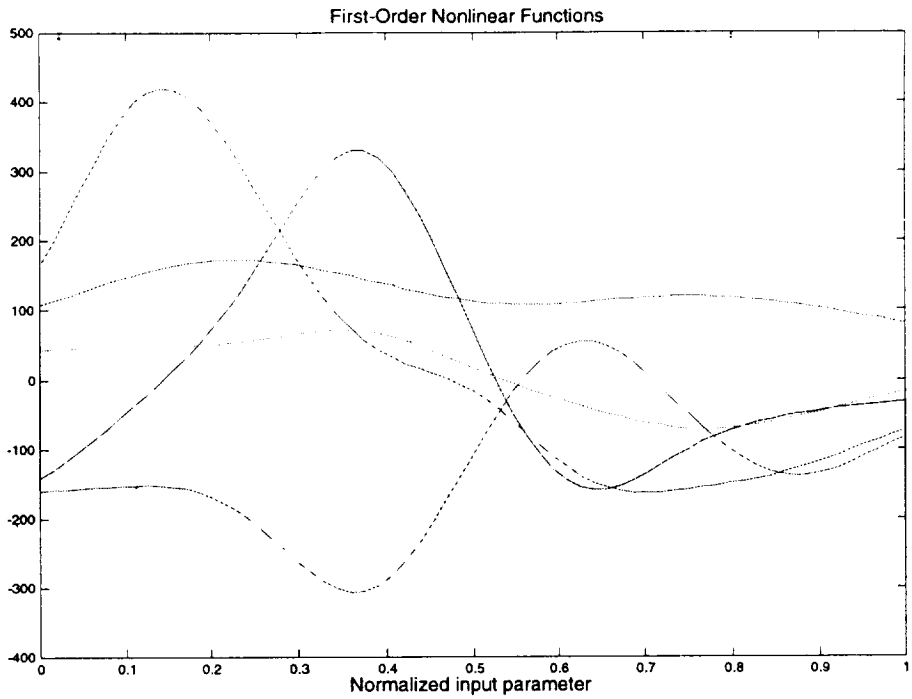

Figure 3: First-order nonlinear functions.

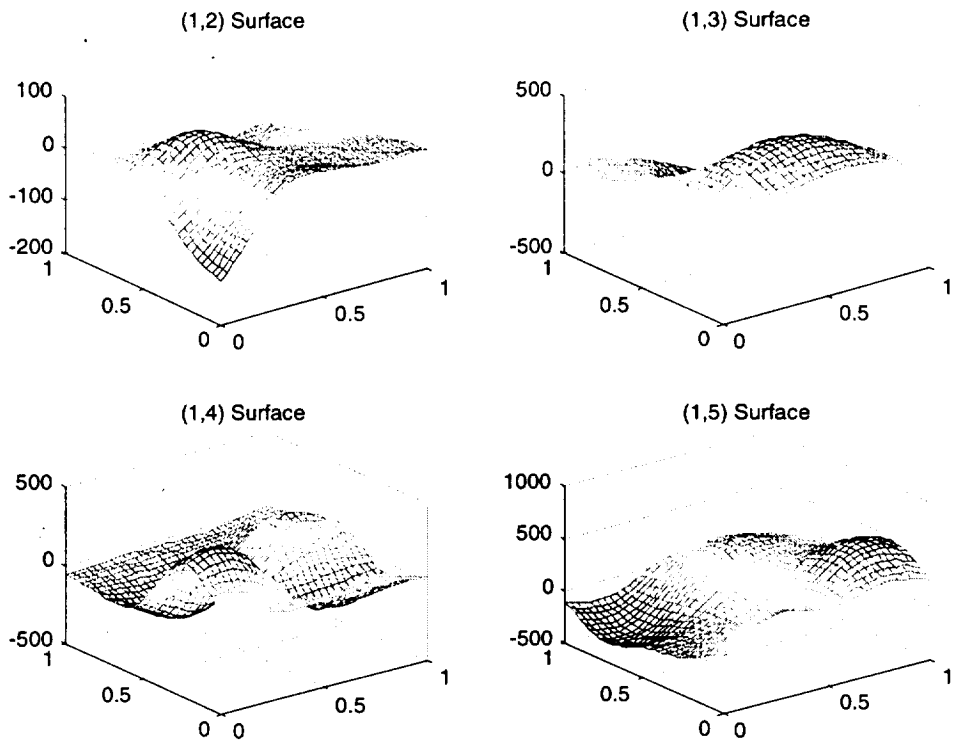

Figure 4: Second-order nonlinear surfaces (1,2)-(1,5).

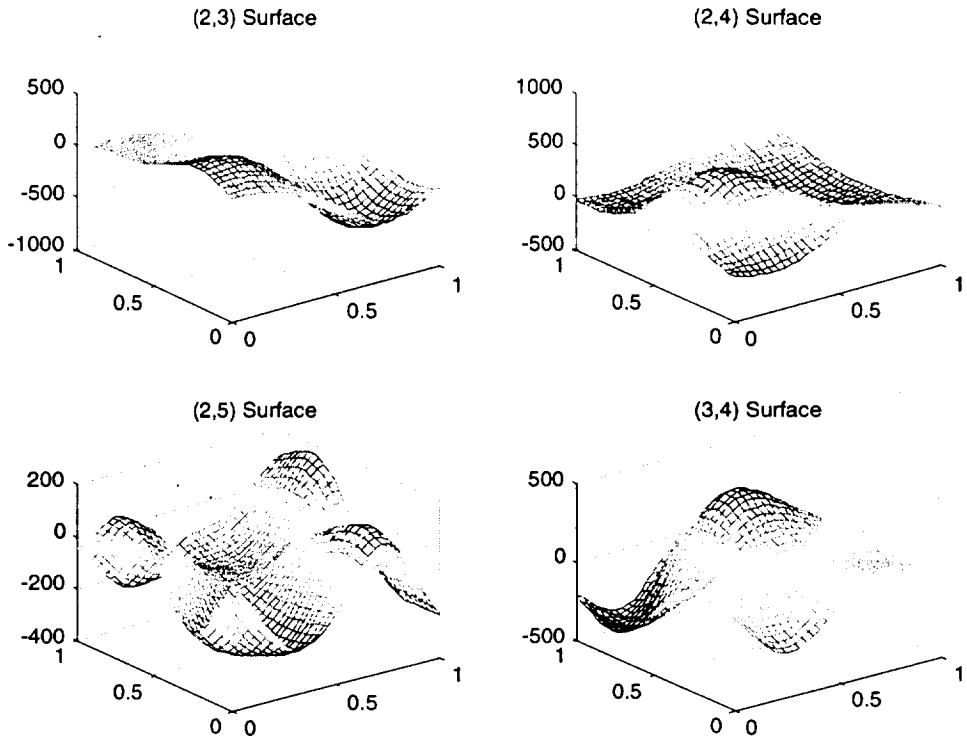

Figure 5: Second-order nonlinear surfaces (2,3)-(3,4).

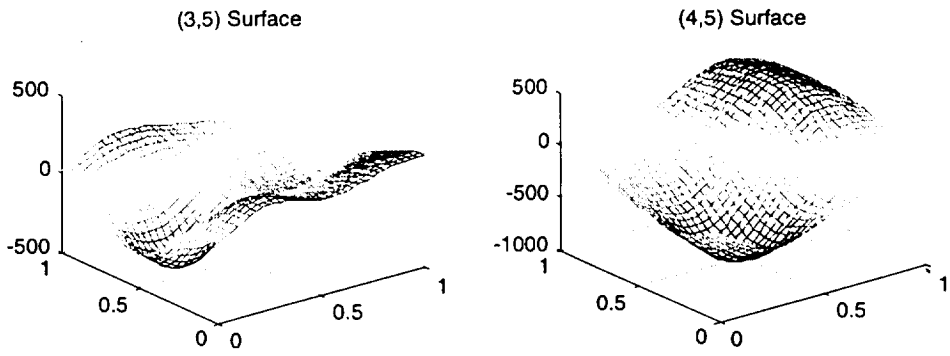

Figure 6: Second-order nonlinear surfaces (3,5)-(4,5).